

# The Design of Fedora

**FUDcon III – London UK**

**October 06, 2005**

# The Importance of Design

- The implementation of good design can always be improved
  - Free Software was a *design* to promote ethical behavior and community among software developers
  - Being liberal in what one accepts and conservative in what one sends is a *design principle* of IETF standards
- The implications of bad design can never be escaped
  - Monolithic vs. modular programs
  - Unpublished proprietary software as a basis for standards
- The book “Massive Change” by Bruce Mau illuminates the design of the world, and thus the world of design
  - See also “Montessori – The Science Behind the Genius”
  - “Unlocking the Sky”
  - “The End of Certainty”

# The Design of Projects

- Good design is invisible – until it fails (Mau)
  - What will the project do?
  - What will the project be?
  - Who will do what?
  - Who needs to know what?
  - Why should anybody care?
- Many projects (alas) confuse lack of design (nonexistence) with good design (invisibility), leading to problems later
- “Nobody goes there anymore—it's too popular”

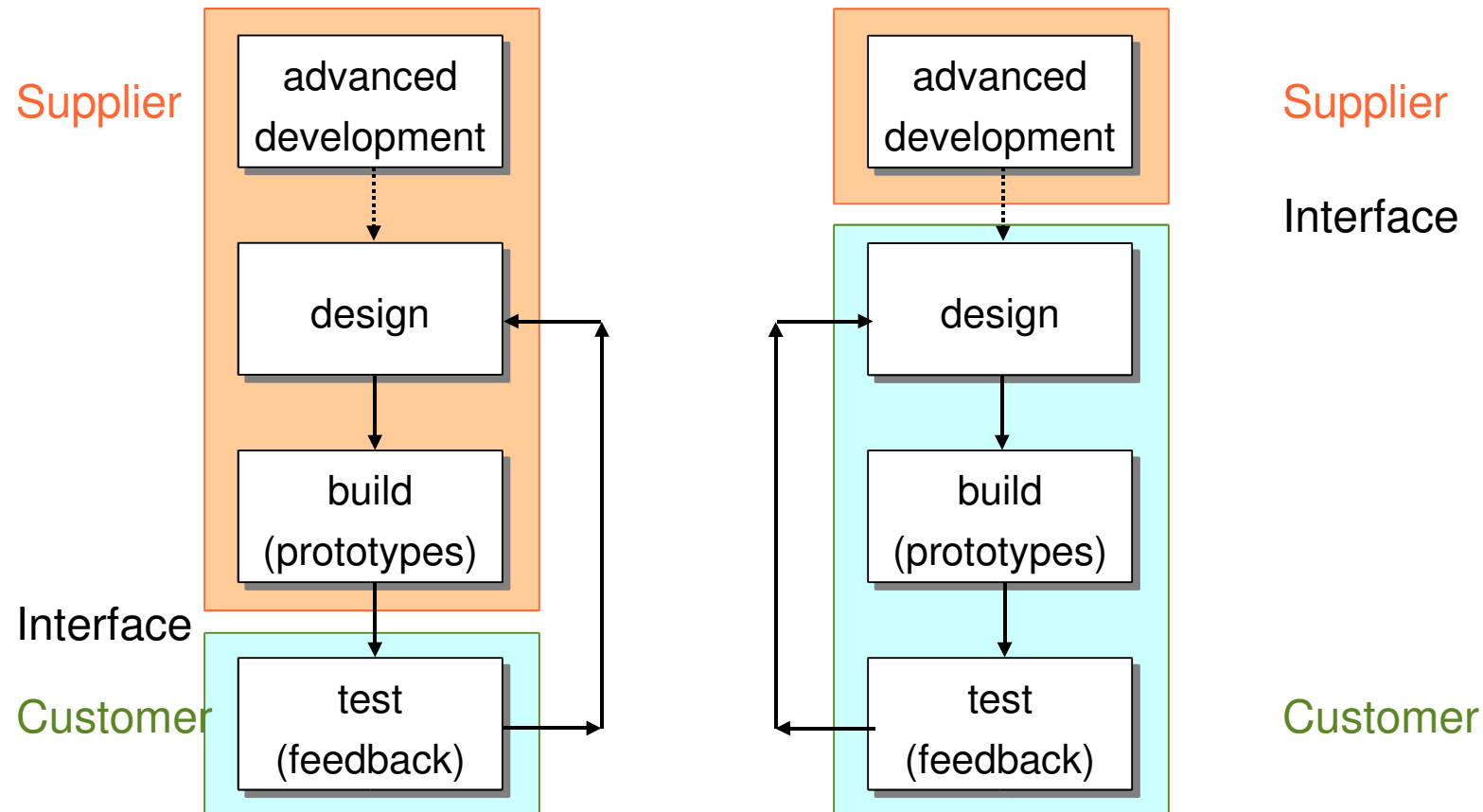
# Upton's Path-based Model

	Installation Based	Path Based
<b>Role of IT</b>	Supportive/Peripheral to Operation	Integral part of Operation
<b>Project Size and Number</b>	Large, few, infrequent	Small, many, frequent
<b>Development Approach</b>	Build, then install	Prototype and evolve
<b>Delivery of Value</b>	When a project is complete	On-going
<b>Source of Technology/ SW</b>	Heavy use of proprietary code, proprietary standards	Standards in common use
<b>Primary Functional Concerns</b>	Control, efficiency, accommodating all requirements at once	Integration, flexibility, progressive delivery of requirements
<b>Locus of Technical Control</b>	Vendor/IT group	Operation itself
<b>Experimentation</b>	Limited	Frequent opportunities

See <http://www.people.hbs.edu/dupton/papers/pathbased-it/PATH.PDF>

Revised May 27, 1997

# User Innovation Toolkits (von Hippel, 2002)



Support for design cycles and “learning by doing”

User-friendly (no new languages or paradigms to learn)

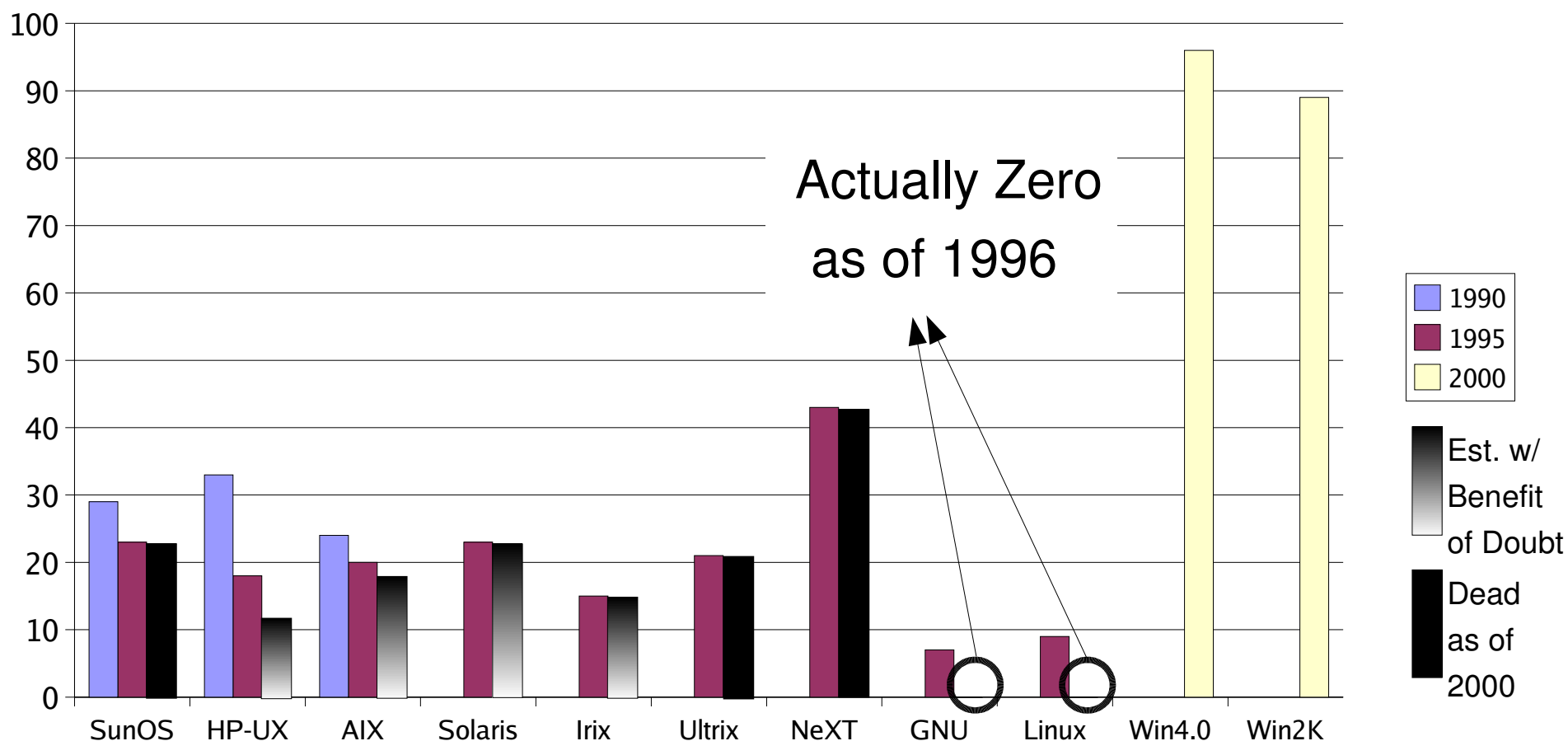
Pretested and debugged libraries

Design Rule checking – the output of the process must be  
“manufacturable”

# Massive Change – Bruce Mau

For most of us, design is invisible. **Until it fails.**

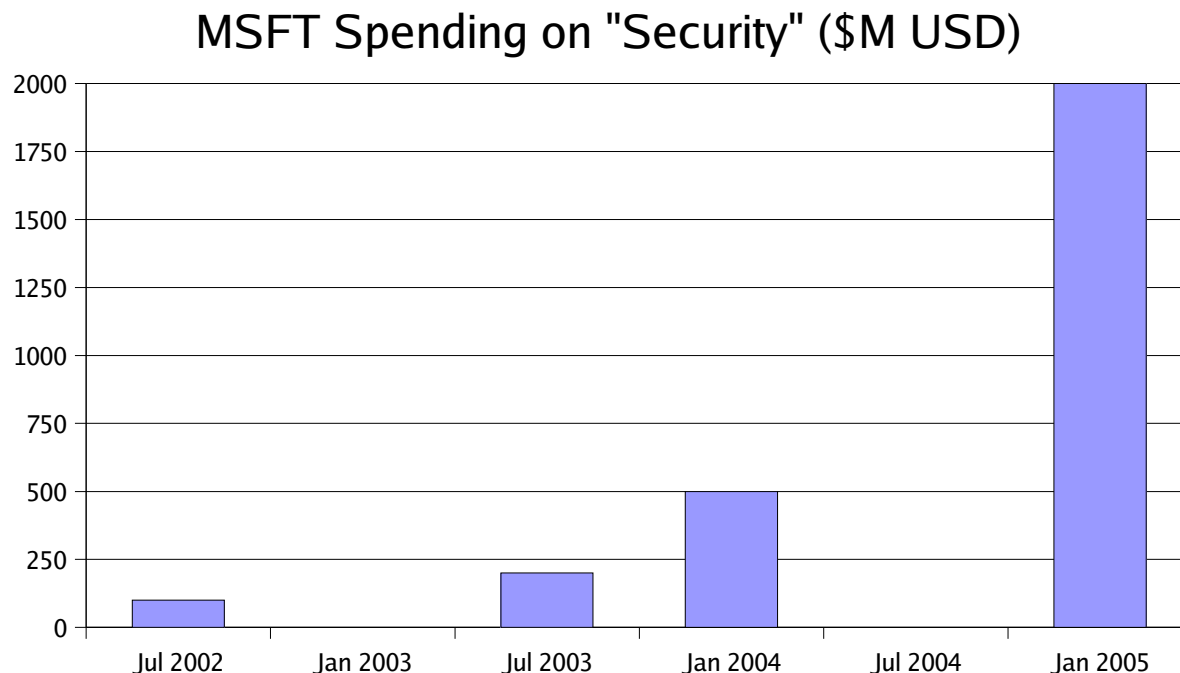
## Fuzz Data 1990-2000



See <http://www.cs.wisc.edu/~bart/fuzz/fuzz.html>

# Massive Change – Bruce Mau

For most of us, design is invisible. **Until it fails.**



July 2002: <http://www.ecommercetimes.com/story/18668.html>

July 2003: <http://www.cnn.com/2003/TECH/biztech/07/17/microsoft.flaw.ap/>

Q1 2004: <http://www.softwarespectrum.com/intouch/edition33/articles.asp?subsection=Microsoft>

Mar 2005: <http://www.fcw.com/fcw/articles/2005/0214/web-gates-02-16-05.asp>

# Design at Red Hat...an ongoing process

- Design has always played a significant role at Red Hat
  - Not just logos...
- Early focus always addressed the questions
  - What was it going to *be*?
  - Why should anybody *care*?
- Examples (good and bad)
  - Red Hat Linux
  - Red Hat Enterprise Linux
  - Fedora Core
  - Fedora Extras



# Fedora Core Design Details

- “Community release” turned out to be too nebulous a design goal
- “A Beta platform for RHEL” was interesting to only a few users
- “Something to replace RHL” was wrong
- But...
  - A 100% pure open source distribution (warts and all)
  - Frequent, roll-forward releases
  - A toolkit for user-driven innovation
  - The best of what works today in the world of open source
  - A new way to approach the development of enterprise-class software
  - ...all had a role in informing design

# Fedora Design sub-goals

- FC1 – Prove we could do it at all
  - Published project goals
  - Published governance strawman
  - Established steering committee
  - Created all-new infrastructure designed to become community-driven
- FC2 – Linux 2.6 kernel, KDE 3.2 and GNOME 2.6
  - More externally than internally focused
  - Proved that Fedora wasn't going away
- FC3 – First tangible input into RHEL
  - SE Linux integrated, “invisible”
- FC4 – Fedora Extras (at long last)

# New Design Considerations

- Use case scenario driven
- Create and/or leverage communities-of-use around core technologies
- Maximize the integration depth obtainable through open source transparency
- Focus on standards and interoperability where applicable

# Implications of Horizontal Computing

- Scalability: Computational and I/O added and removed as needed
- Reliability: Failures of systems do not cause application downtime
- Cost: Commodity, industry standard solutions
- Flexibility: Resources easily re-purposed as needed for other applications
- Performance: New innovations can be introduced rapidly (multi-core, DDR2)

tomorrow's building block for  
horizontal computing



64-bit address space

Out-of-band management  
capabilities

4-socket x 4 cores X HT = 32-  
way

Virtualization and security  
assistance

# Architectural Design

- Disaggregation of applications, servers, and storage
  - Virtualization touches all levels
- Always On, Continuous Availability
- Development platform impact on cost and opportunity:  
performance/security/quality/analysis/time-to-deploy
- Seamless security of applications, services, instances, and access
- Paradigm shift to service level management rather than managing components
  - Integrated management of all elements

# Evolution of the Development Platform

Foster a federated development platform with unlimited scale,  
reach, and participation.

- Enabling large-scale communities of use and innovation
- Enable and support the ability for participation in the reach of OSS
- Help developers, customers and partners direct involvement to shape their future
- Definition of development tools and worldwide infrastructure needed for:
  - SCM
  - Test automation
  - Test and development platforms
  - Developer tools

# Looking forward

- FC4 has some bits for new architecture of computing
  - Xen
  - GFS
  - FDS
  - ...but existence != integration
- FC5 will hopefully *demonstrate* new use-case scenarios
  - Controlled and uncontrolled shutdown
  - Desktop grid
  - Management of many virtual machines on a single box
  - Architecture independence (for appropriate apps)
  - ...with your help as users and developers